Partitioning Contact-State Space using the Theory of Polyhedral Convex Cones

George V Paul and Katsushi Ikeuchi

CMU-RI-TR-94-36

Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

August 1994

This work was sponsored in part by National Science Foundation under IRI-9224521 and in part by the Avionics Laboratory, Wright Research and Development Center, Aeronautical Systems Division (AFSC), U.S. Air Force, Wright-Patterson, AFB, OH 45433-6543 under F33615-90-C-1465 and F33615-93-1-1282.

Keywords: Assembly planning, Robot programming, Teach-by-showing, Teleoperation, Image Understanding, Object Recognition, Task Recognition, Screws.

Abstract

The assembly plan from observation (APO) system observes a human operator perform an assembly task, analyzes the observations, models the task, and generates the programs for the robot to perform the same task. A major component of the APO system is the task recognition module, which models the observed task. The task model in the APO context is defined as a sequence of assembly states of the part being assembled and the actions which cause the transition between states. The state of the assembled part is based on its freedom which can be computed from the geometry of the contacts between the part and its environment. This freedom can be represented as a polyhedral convex cone (PCC) in screw space. We show that any contact configuration can be classed into a finite number of contact states. These contact states correspond to topologically distinct intersections of the PCC with a linear subspace T in screw space. The models of any observed task can be represented compactly as a path in a transition graph obtained from these contact states. We illustrate the application of the theory by implementing the APO system for the case of objects assembled in a plane using rotation and translation motion.

Table of Contents

List of Figures

1 Introduction

Conventional methods of programming and operating robots are teach-by-showing, teleoperation, textual programming, and automatic programming. This work is based on a system which is based on a novel method of programming robots called Assembly Plan from Observation (APO)[23].

When we consider a task such as assembly, the conventional methods all have drawbacks. Teach-by-showing [11] has the disadvantage that the control is low-level, typically at the joint level. Hence it is not robust enough for programming robots for assembly tasks. Textual programming [3] needs a programmer trained in specialized robot programming languages such as VAL, who converts the assembly task into basic robot commands which can accomplish the task. Teleoperation [4][21][26] needs a human operator as an essential part of the system. So, a system based on it cannot operate autonomously. Unlike the previous methods, automatic programming [13][17][14] needs no human intervention after the specification of the task. The system can identify and compute the solutions for all aspects of the task automatically. But the disadvantage of this method is the overwhelming complexity of the subtasks.

The APO system observes a human perform an assembly task before a perception system. It then understands what the human did and how it was done. This information is used to program a robot. Thus it incorporates the simplicity of teach-by-showing and teleoperation. In addition, the robot programs are generated automatically. By using the operators solutions for the subtasks like assembly planning, grasp positioning, and collision free path planning instead of computing them, the APO system avoids the complexity of automatic robot programming.

When a part is assembled into another, there are three major aspects of the task that can be observed. They are, the collision free path from the parts table to the assembly, the fine motion resulting in the assembly, and the grasping strategy used to manipulate the part. In this work, we concentrate on the fine motion strategy used by the operator to bring the part into its final configuration.

To understand what was done and how it was done, the APO system observes critical contact states of the manipulated part. It then uses models of tasks and matches the observed task to one of the task models. The basic task model consists of a pre-assembly state, a post-assembly state and the action necessary to get from the former to the latter. To represent all possible assembly tasks, we classify all the possible contact configurations of the manipulated part into a finite number of assembly states. The contact states are used to build a transition graph. The model of the observed task forms a subgraph in this transition graph. Once this abstract task model of the observed task is obtained, we use motion templates to generate the manipulator motions needed to program the robot manipulator to perform the task. The APO system has already been implemented for polyhedral objects when the assembly motions are all translations [23].

Although the previous APO system can handle a significant variety of assembly tasks, it cannot reason about assemblies involving rotations. We have extended the APO system such that it can deal with assemblies of polygonal objects that are assembled using rotation in addition to translation.

We use screw theory to represent the freedom of a contact configuration of the assembled object. We then use the theory of polyhedral convex cones to classify the contact inequalities into a finite number of topologically distinct states. The finite states are then used to build a transition graph. The representation of the freedom and its classification based on topology results in a comprehensive theory for partitioning contact state space.

Among related work, the most closely related is the work of Hirai [5][6], which deals with the kinematic analysis of contacts to model robot assembly tasks based on contact states. Laugier [12] also uses the contact geometry to plan fine motion during assembly. Other work such as by Mattikalli [18] deal with representation of the motion constraints based on contact geometry. The concept of partitioning contact state space and using it to build task models was first proposed by Ikeuchi [23]. Our work provides the theoretical basis for this.

1.1 Organization of this Report

Section 2 introduces the components of the system and the connections between them. It also introduces the concepts involved in building the task recognition module for planar assembly using translation only. Section 3 details the building of the task recognition module for planar assembly with rotation and translation. This section contains most of the contributions of this work. Section 4 explains the implementation of the system for polygonal objects assembled in a plane. The final section draws conclusions on this work and discusses future plans.

2 The Assembly Plan from Observation (APO) System

The aim of this chapter is to briefly introduce the components of the APO system. The schematic of the APO system is shown in Figure 1. The three major components of the APO system are the observation module, the task recognition module and the task instantiation module. We will indicate the input, output, and the method of operation of each component.

The human operator performs the assembly task by manipulating parts in a sequence of subtasks. The observation module observes the operator perform each subtask. The task recognition module then understands what was done, how it was done, and uses subtask models to represent the observed subtasks. Finally, the task instantiation module converts the modelled subtasks into a task for the robot. This instantiated task is then used to generate the program that will enable the robot to perform the assembly task in its workspace.

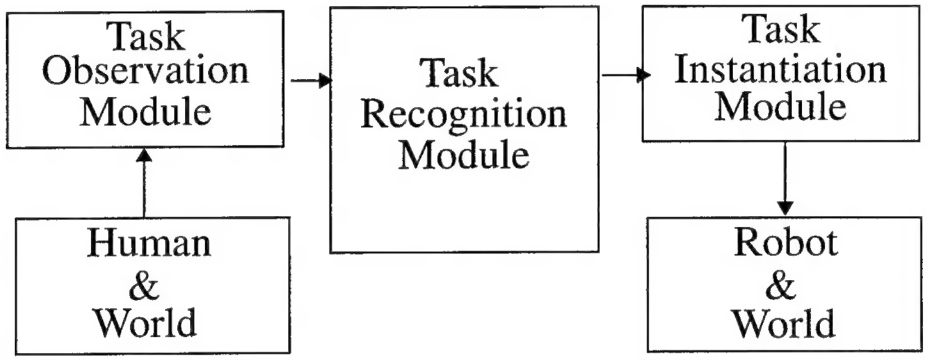


Figure 1 : Assembly Plan from Observation System

2.1 Observation Module

The observation module uses a perception (presently, vision only) system to observe the human operator performing the assembly task. The observation module performs the following:

- Temporal Segmentation - The vision sensors of the perception system provide us with a continuous sequence of images of the operator performing the assembly. These have to be split into meaningful segments in time. A meaningful segment will correspond to one assembly subtask. Figure 2 shows the temporal segmentation of a planar assembly task into 5 segments corresponding to 5 subtasks.

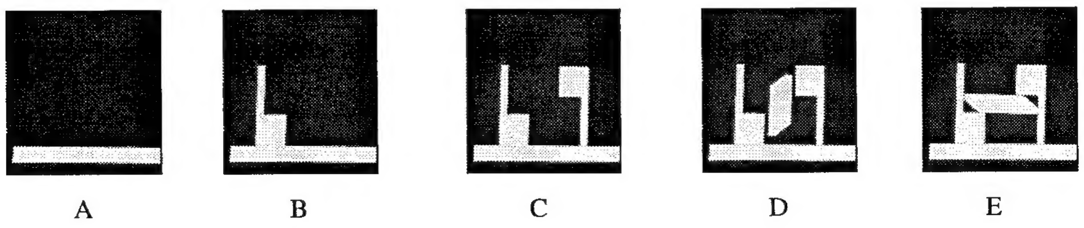


Figure 2 : Temporal segments of a planar assembly task

- Object Recognition - In order to understand what the operator achieved during a stage of the assembly, the system has to identify the object that was manipulated. The object recognition program identifies the manipulated object and its configuration with respect to a world coordinate system.

2.2 Task Recognition Module

In each stage of the APO system operation, the operator manipulates an object (called the manipulated object) onto the previously assembled objects in the scene (called the environmental objects). The aim of the task recognition module is to understand what was done by the operator and how it was done. This involves the following:

- Modelling the Assembly Subtask - Every assembly subtask involves at least one change in the assembly relations (usually contacts) between the manipulated and environmental objects. Using the geometrical models of the objects and the configuration of the objects, it is possible to compute this change in assembly relations and identify the actions that caused it. This is called modelling the assembly subtask. The complete assembly task can be represented as a series of such subtask models.
- Grasp Recognition - While performing the assembly task, it is useful to understand where and how the operator grasps the assembled objects. This can be used for planning the grasp for the robot. A detailed description of grasp recognition from observation can be found in [8].
- Global Path Recognition - The objective here is to observe the path of the manipulated object used by the operator performing the assembly. Using the path taken by the operator avoids the computation needed for searching out a collision free path in configuration space [16].

The major part of this work deals with the theory behind modelling assembly subtasks which are defined by contacts. We will try to illustrate the theory of task modelling based on contacts, for the simple case where polygonal objects are assembled in a plane using only 2D translation motions.

2.2.1 Assembly Subtask Modelling

If the configuration of the objects forming an assembly are known, we can geometrically deduce the assembly relations (contacts) established between the manipulated object and the environmental objects. Each of these contacts constrains the motion of the manipulated object. The set of contacts in an assembly relation restrict the motion of the assembled object to a set of legal motions. The actions that result in the assembly must be a member of this set of legal motions.

The major premise of the APO system is that the actions of the operator while performing the assembly are aimed at causing the change from one assembly relation to another. Using assembly relations as the basic representations, an assembly subtask can be defined as involving a pre-assembly relation, a post-assembly relation and the action that causes the transition from the former to the latter (Figure 3).

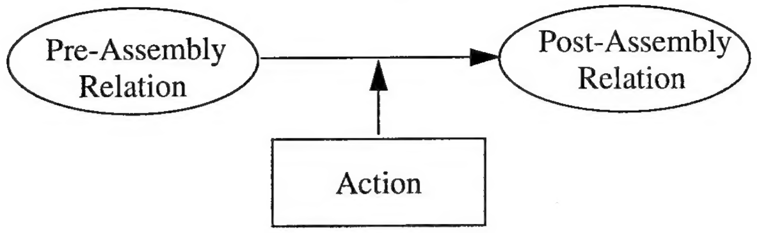


Figure 3 : Task model schematic

There are infinite number of possible assembly configurations and actions which can cause the transition from one to another. We represent them using a finite number of task models. This involves the following steps.

- Classifying all possible contact relations (assembly relations) between the manipulated and the environmental objects into a finite number of contact states.
- Building the transition graph by finding all the transitions that can occur from the pre-assembly state to the post-assembly state. Each arc of the graph corresponds to one possible transition and each node corresponds to an assembly state.
- Assigning manipulator actions to achieve such assembly state transitions (the completed graph is referred to as a procedure graph).

2.2.2 Subtask Modelling for Planar Assemblies: Translation only

The following subsections describe the concepts and the methods involved in building the subtask modelling for the assembly of polygonal objects. This is restricted to a two-dimensional world where actions can be translations only. Section 3 will describe the building of the subtask modelling module for the two-dimensional world with rotation in addition to translation.

2.2.2.1 Assembly Relation

Consider two objects in contact as shown in Figure 4 (a). The contact relation can be completely defined by the surface normal $\boldsymbol{n}$ at the points of contact. The translation vector $\boldsymbol{t}$ which represents detaching or sliding of the two bodies, can be defined by the inequality (1).

$$\boldsymbol{n} \cdot \boldsymbol{t} \geq \boldsymbol{0} \tag{1}$$

The solution for (1) can be represented by the unshaded semicircle on a gaussian circle as shown in the Figure 4 (b). Any vector on the unshaded semicircle will cause the objects to detach. Any vector on the two ends of the semicircle will result in sliding of the objects. The inequality (1) forms the basis of the representation of assembly tasks

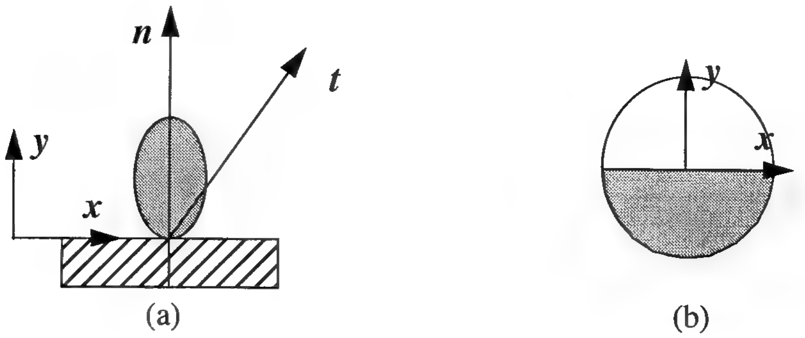


Figure 4 : (a) Single contact geometry. (b) Assembly Freedom representation

2.2.2.2 Classification of Assembly Relations

When there are multiple contacts, then (1) becomes a system of inequalities given by:

$$N \cdot \boldsymbol{t} \geq \boldsymbol{0} \qquad (2)$$

N is the matrix of contact normals. The solution space of (2) can be represented as the intersection of the semicircles on the gaussian circle as shown in Figure 5. The solution of (2) represents the freedom of one object with respect to the other. This freedom can be classified into a finite number of distinct contact states based on the topology of the solution space. This is based on the theory of polyhedral convex cones as will be explained in Section 3. In the 2D planar case with translation only, this will result in six contact states which are described below and are shown in Figure 5.

State S: There are no contacts and the object motion vector, $\boldsymbol{t}$ can be any 2D vector.

State A: There is a single direction of contact, so the solution can lie only on a semicircle.

State B: There are two non-parallel contact normals, and the rank of N is two. The solutions can lie on a sector bounded by the two non-parallel vectors.

State C: The two contact normals are anti-parallel and the rank of N is one. The solution can lie on a line.

State D: Three contact normals. A third contact normal, in addition to the two in state C. The solution will be a single direction of translation.

State E: The contact normals are such that there is no feasible solution.

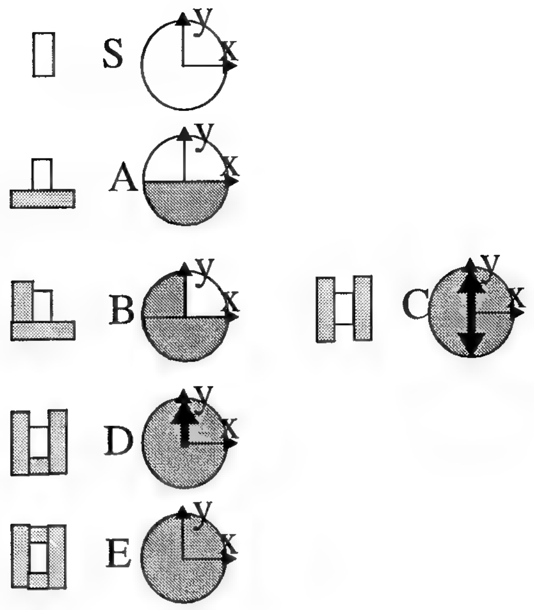


Figure 5 : Contact States for 2D Translation.

2.2.2.3 Transition Graph for Polygonal objects

The goal of the assembly task is to change the contact state of the object. The transition graph represents all the possible transitions between contact states and the actions necessary to achieve them.

Theoretically there can be 5^2 transitions between the 5 states. But if we consider only transitions which increase the number of contacts gradually and also assume polygonal objects, we obtain 5 transitions between the 5 states. The transition graph is shown in Figure 6(a).

When the APO system is in operation, the task recognition module maps the observed object configurations, into contact states in the transition graph. It also maps the observed actions into the transitions between the states in the graph. The states along with the transitions are used to represent each observed assembly subtask.

The transition graph is the abstract representation of all the possible assembly tasks. In order to use these models to program a robot, we need the manipulator actions necessary to achieve each transition. The actions corresponding to each transition is called a motion macro. These motion macros can be used to program the robot.

When all the transitions in the transition graph are associated with its corresponding motion macro, the resulting graph is called a procedure graph (See Figure 6(b)). M_1, M_2. M_5 are the motion macros for corresponding to the 5 transitions.

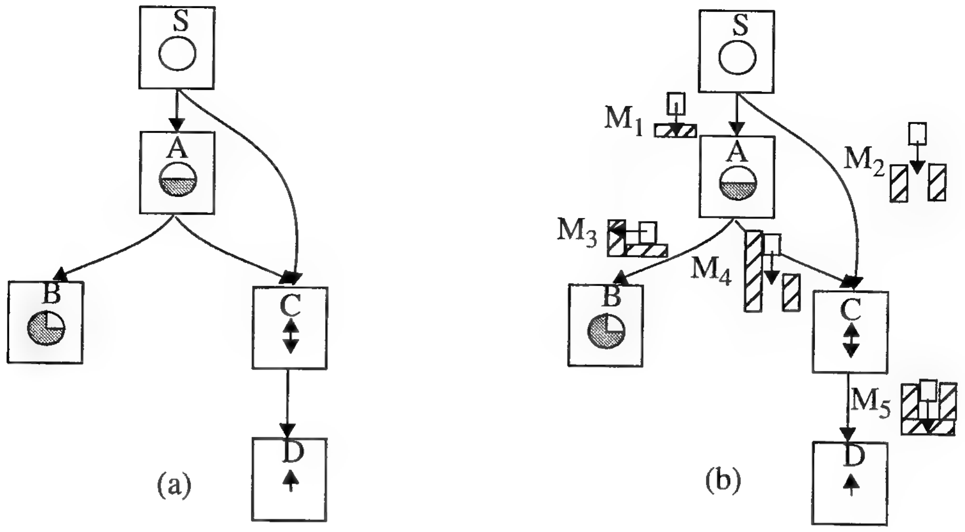


Figure 6 : (a)Transition Graph (b) Procedure Graph.

2.3 Task Instantiation Module

The task recognition module represents the observed task as an abstract task model. This abstract task model has to be converted into a task for the robot. This is done by the task instantiation module.

Here the subtask model is used to generate the programs for the robot to perform the assembly subtask in its workspace. In order to accomplish this, additional information of parameters like the distance of translation are needed. These can be obtained from the results of object recognition, grasp recognition, and global path recognition. The instantiated task can be used to generate the program for the robot. The task instantiation module can also incorporate skills like hybrid control and visual feedback, to increase robustness during task execution.

The subtask modelling part of the APO system for 2D polygonal objects described above serves as an introduction to the concept. The detailed building of the APO system for the case of 2D rotation and translation is described in Section 3

3 Subtask Modelling for Planar Assemblies: Translation and Rotation

Assembly tasks that require both rotation and translation cannot be analyzed using the surface normals of contacts on a gaussian circle. Instead, we use screw theory, which can represent rotation and translations in a single mathematical framework [1][7].

The screw is a convenient concept for representing three dimensional rigid body displacements [1]. Any rigid body displacement can be accomplished by a rotation about a unique axis and a translation along this same axis. The combined motion is called a screw displacement or twist. The unique axis is referred to as the screw axis, and the ratio of the translation to the rotation is designated as the pitch of the screw. The amount of rotation about the screw axis is called the amplitude of the screw.

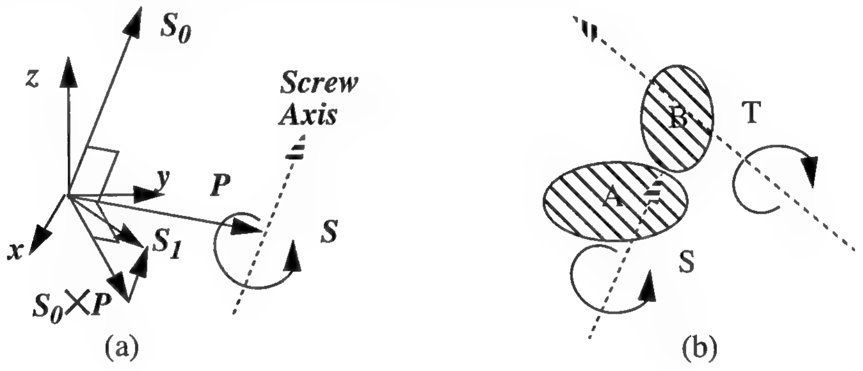


Figure 7 : Screw coordinates

A screw ***S*** is represented mathematically by two 3D vectors, $[S_0, S_1]$ as shown in Figure 7(a). S_0 is the direction of the screw axis and $S_1 = S_0 \times P + pS_0$. The vector $S_0 \times P$ is the moment of the screw axis about the origin, ***P*** is a vector to the screw axis from the origin and the scalar *p* is the pitch of the screw.

For pure rotations, the pitch $p=0$, and the screw ***S*** will be $[S_0, S_0 \times P]$. For pure translations, the pitch *p* is infinity, so the screw ***S*** will be $[0, S_0]$. Screws can also be used to represent a line in space. In this case, the screw axis will be along the line, and the pitch of the screw will be zero [1][7].

3.1 Assembly Relation

Consider two bodies in contact as shown in Figure 7(b). Let screw $S = [s_1, s_2, s_3, s_4, s_5, s_6]$ represent the line of contact in 3D space and screw $T = [t_1, t_2, t_3, t_4, t_5, t_6]$ represent the displacement of one object with respect to the other. Any screw ***T*** that can cause the sliding of object B on object A is called a reciprocal screw and is related to ***S*** by equation (3) as explained in [1].

$$s_1t_4 + s_2t_5 + s_3t_6 + s_4t_1 + s_5t_2 + s_6t_3 = 0 \tag{3}$$

Any screw ***T*** that can cause the detaching of object B from object A is called a repelling screw and is related to ***S*** by the inequality (4) as explained in [19].

$$s_1t_4 + s_2t_5 + s_3t_6 + s_4t_1 + s_5t_2 + s_6t_3 > 0 \tag{4}$$

Thus the relation that defines the feasible set of legal motions which do not violate the contact (sliding and repelling) can be written as the inequality (5) [19]. This is called the contact inequality.

$$s_1t_4 + s_2t_5 + s_3t_6 + s_4t_1 + s_5t_2 + s_6t_3 \geq 0 \tag{5}$$

The contact inequality can represent any contact between objects in 3D space. If we consider two bodies in contact in a plane as shown in Figure 8(a), then the screw representing the contact will be $[s_1, s_2, 0, 0, 0, s_6]$, where $s_1=n_x$ and $s_2=n_y$ and $s_6=(p_yn_x - p_xn_y)$. The vector (n_x,n_y) is the contact normal, and (p_x,p_y) is the vector from the origin to the line of contact. ***T***$=[0, 0, t_3, t_4, t_5, 0]$ will represent the screw displacement in the plane. The inequality representing any feasible displacement in the plane can then be given as (6).

$$at_x + bt_y + cw_z \geq 0 \tag{6}$$

where $a = n_x$, $b = n_y$ denotes the contact normal and $c = (p_yn_x - p_xn_y)$. The variables $t_x = t_4$, $t_y = t_5$ and $w_z = t_3$ represent the screw displacement in the plane.

The screw displacement in a plane represents either a counter-clockwise rotation or a clockwise rotation, about some rotating center in the plane. If the solution of (6) is $[t_x, t_y, w_z]$, and w_z is positive, then it implies a counterclockwise rotation. If w_z is negative, it represents a clockwise rotation. The freedom of an object in contact can be represented by the feasible rotation centers for counter-clockwise and clockwise rotations.

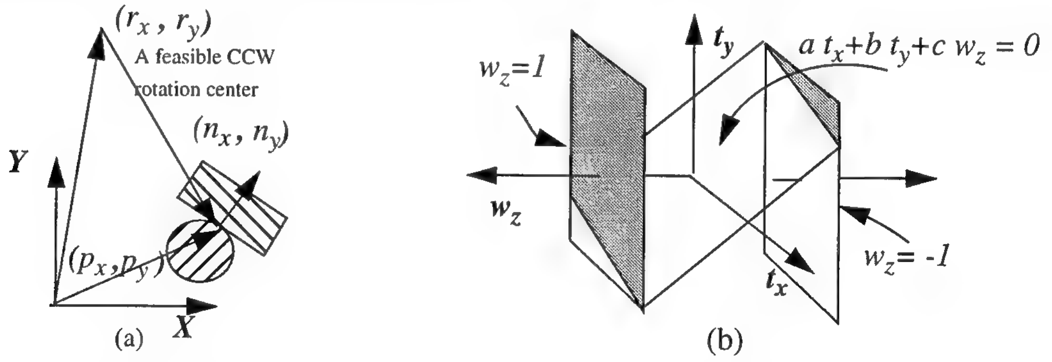


Figure 8 : (a) Contact Geometry (b) Constraint Space for 2D Translation and Rotation.

3.1.1 Graphical Representation of the Freedom of an Assembly Relation

Using the contact inequality it is possible to find the areas where the feasible rotation center areas for clockwise and counter-clockwise rotations can lie. These areas can be used as an intuitive representation of the freedom of the assembly relation.

Consider the coordinate space $[t_x, t_y, w_z]$, the solution of the contact inequality (6) will be represented as a half-space. The half space will be bounded by a plane passing through the origin as shown in the Figure 8(b). The solution space corresponding to counterclockwise and clockwise rotations will then be intersections with the two planes $w_z = -1$ and $w_z = +1$ as shown in the Figure 9(a).

Given a feasible solution $[t_x, t_y, w_z]$, $w_z = \pm 1$, and assuming that the corresponding rotation center is $[r_x, r_y]$. If $w_z = +1$, the rotation center for counterclockwise rotations is given by equation (7). If $w_z = -1$, the rotation center for clockwise rotations is given by equation (7).

$$[t_x, t_y, w_z] = [-r_y, r_x, +1] \tag{7}$$

$$[t_x, t_y, w_z] = [r_y, -r_x, -1] \tag{8}$$

Figure 9(b) shows how the solutions of the inequality are mapped onto the plane of the bodies in contact as corresponding rotation centers. All points to the one side of the contact line can be feasible centers for counterclockwise rotation, and all points to the other side of the line of contact can be feasible centers for clockwise rotation. Any rotation centers on the contact line will maintain the contact. These points correspond to the feasible solutions lying on the constraint plane $(a\, t_x + b\, t_y + c\, w_z = 0)$.

The rotation center areas can be used to graphically represent the freedom of the contact as shown in Figure 9(c). The shaded regions on the left and right rectangles indicate the feasible counterclockwise and clockwise rotations respectively.

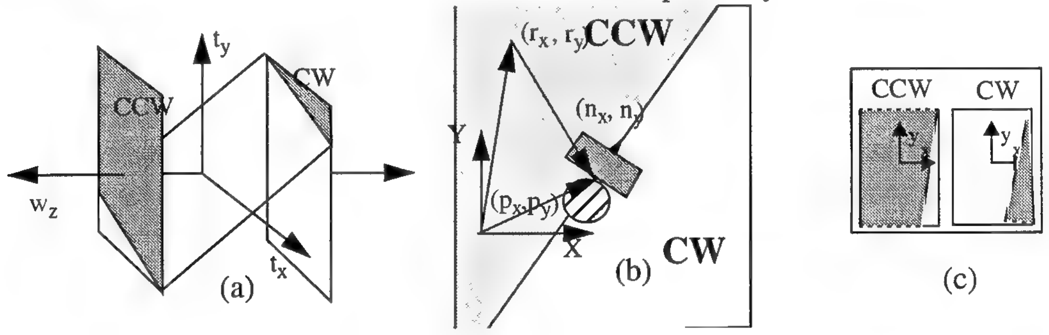


Figure 9 : (a) Feasible Solutions (b) Rotation Centers (c) Representation

When there are multiple contacts, the contact inequality (6) becomes a system of inequalities as shown in (9).

$$\begin{vmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ \dots & \dots & \dots \\ a_m & b_m & c_m \end{vmatrix} \begin{bmatrix} t_x \\ t_y \\ w_z \end{bmatrix} \geq \begin{bmatrix} 0 \\ 0 \\ \dots \\ 0 \end{bmatrix} \tag{9}$$

The solution of (9) will be the intersection of the half-spaces corresponding to each inequality. This solution space is called a polyhedral convex cone (PCC) in $[t_x, t_y, w_z]$ space. The feasible rotation centers will correspond to the intersection of this polyhedral convex cone with the $w_z = \pm 1$ planes.

Consider the case of two contacts as shown in Figure 10(a). The polyhedral convex cone is a wedge-like space as shown in Figure 10(b). Feasible rotation centers exist only for counter-clockwise rotations as shown in Figure 10(c). The freedom of the object is very different from the single contact case considered earlier. The object in this case can only translate in two directions which lie on a line.

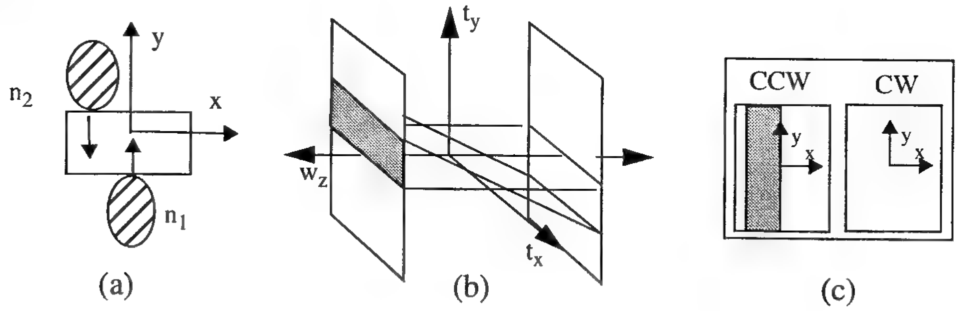


Figure 10 : Two Contact Case

The freedom of the object in the single-contact case and the two-contact case are obviously different. They are said to belong to distinct contact states. The infinite number of possible contact configurations[1] can be classified into a finite number of contact states. The intuitive reason why two contact states are different is that their rotation center areas are different. The precise explanation using the theory of polyhedral convex cones is described in the following section.

3.2 Contact States

A graphical representation of the feasible rotation center areas gives us an idea of the different types of freedoms that can result from multiple contacts. In order to systematically generate all the possible contact states, we use the topology of polyhedral convex cones.

3.2.1 Polyhedral Convex Cones (PCC)

Consider the system of m linear inequalities $\boldsymbol{AX} \geq \boldsymbol{0}$ shown in (10). $\boldsymbol{X}$ is a n-dimensional vector. The solution space of $\boldsymbol{X}$, denoted by $\boldsymbol{A^*}$ will be a polyhedral convex cone (PCC). The theory of polyhedral convex cones is explained in the paper by Tucker and Goldman in [10].

$$\begin{vmatrix} a_{11} & \dots & a_{1n} \\ a_{21} & \dots & a_{2n} \\ \dots & \dots & \dots \\ a_{m1} & \dots & a_{mn} \end{vmatrix} \begin{bmatrix} x_1 \\ \dots \\ x_n \end{bmatrix} \geq \begin{bmatrix} 0 \\ 0 \\ \dots \\ 0 \end{bmatrix} \qquad (10)$$

The solution of each inequality in (10) is represented by a half space in n-dimensional space bounded by a face of dimension *(n-1)*. The solution to (10) is the intersection of the m half-spaces and will form a PCC. The PCC is defined topologically by the number and

1. We use vertex-on-line contacts to represent the contact configurations. Line-on-line contacts which are common in real life assemblies can be represented by two vertex-on-line contacts situated at the ends of the contact line [19].

dimension of the faces constituting it. The faces[2] of a PCC in n-dimensional space can be subsets of the n dimensional space, $(n-1)$ dimensional space, ... , 0 dimensional space.

If n is the dimension of $\boldsymbol{X}$ and r is the rank of matrix $\boldsymbol{A}$, then d, $(d=n-r)$ is the smallest dimension of the faces constituting the solution space $\boldsymbol{A}^*$. This face is called the *d-face*. Theorem-1 in [10] states that the solution $\boldsymbol{A}^*$ is either a linear subspace of dimension d, or the convex hull of finitely many half subspaces of dimension $(d+1)$ bounded by a common subspace of dimension d.

A consequence of this theorem is that the PCC $\boldsymbol{A}^*$ can be composed of faces of dimension $d, d+1, ..., n$. Topologically distinct solutions $\boldsymbol{A}^*$ arise when the dimension of faces constituting $\boldsymbol{A}^*$ belong to the sets $\{d\}$ or $\{d, d+1\}, ..., \{d, d+1, ... n\}$. In addition, for PCCs in n-dimensional space, the value of d can vary from 0 to n. Using these facts we can generate all possible topologically distinct PCCs in a n-dimensional space.

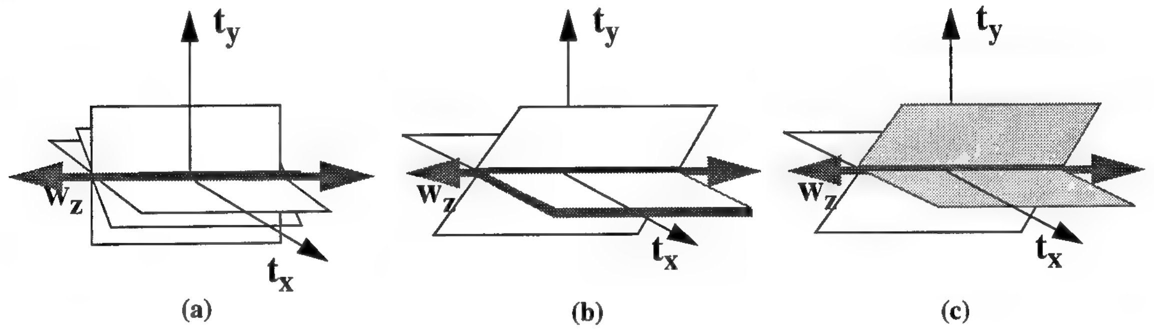


Figure 11 Polyhedral convex cones in 3d space

An example of the three topologically distinct solutions $\boldsymbol{A}^*$ predicted by the theorem, when $n=3$, $r=2$ and $d=1$ is shown in Figure 11. Figure 11(a) shows the case when the solution $\boldsymbol{A}^*$ is purely a linear 1D subspace (a line). Figure 11(b) shows the case when the solution $\boldsymbol{A}^*$ is composed of a linear 1D subspace (a line) bounding a 2D half subspace (half-plane). Figure 11(c) shows the case when the solution $\boldsymbol{A}^*$ is a convex cone composed of a linear 1D subspace (a line) bounding two 2D half subspaces (half-planes), which bound the 3D space.

When we apply the theory of PCC to the 3D space $[t_x, t_y, w_z]$, which represents the freedom of a body in a plane as explained in Section 3.1, the system of inequalities resulting from the contacts on an object in a plane will be given as (9).

$$\begin{vmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ \dots & \dots & \dots \\ a_m & b_m & c_m \end{vmatrix} \begin{bmatrix} t_x \\ t_y \\ w_z \end{bmatrix} \geq \begin{bmatrix} 0 \\ 0 \\ \dots \\ 0 \end{bmatrix} \qquad (11)$$

2. The faces of a PCC in a three dimensional space can be subsets of a point (zero dimensional face), a line (one dimensional face), a plane (two dimensional face) or a 3D space (three dimensional face)

The solution A^* will now be a PCC in 3D space. The topologically distinct solutions will form the basis for classifying contact configurations into a finite number of contact states. For example, when A^* is a 2D plane bounding a 3D space, the corresponding contact configuration is a single contact (see Figure 12(a)). When A^* is just the 2D plane, it corresponds to two opposing contacts lying on the same line (see Figure 12(b)).

Topologically equivalent solutions can correspond to different contact states. The reason for this is as follows. The coordinates t_x and t_y are different from the coordinate w_z. The translation freedom depends on the intersection of the solution space A* on the t_xt_y planes. therefore, if the same PCC intersects the t_xt_y plane in n different ways, they will correspond to n contact states (see Figure 13).

Topologically Distinct Solutions

We will now systematically analyze all the possible contact states in the following sections. For the (t_x,t_y,w_z) space, the *d-face* can be the whole 3D space, a 2D plane, a line, or a point. The faces other than the *d-face* can be subsets of spaces of dimensions $d+1$ to 3.

The *d-face* is the 3D space

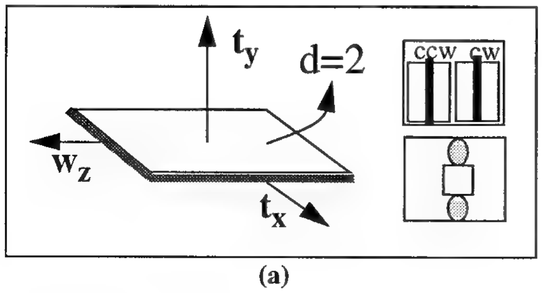
When the *d-face* is the entire 3D space there are no contacts with the object. The object is free to rotate and translate in any direction.

The *d-face* is a Plane

When the rank of A is 1, the *d-face* has dimension $d=2$. The solution A^* according to the theorem will be either a linear 2D plane or a 3D half space bounded by a 2D plane.

The first case and its physical contact configuration is shown in Figure 12(a). Here, the intersection of A^* with the t_x-t_y plane can be either a line or the whole t_x-t_y plane. The former case is shown in Figure 12(a). The latter case is physically impossible because it corresponds to contacts located at infinite distance relative to the origin.

Figure 12(b) shows the case when the solution A^* is a 3D half space bounded by a 2D plane. No other cases due to the intersection with the t_x-t_y plane occur because the case when the *d-face* (the 2D plane) lies in the t_x-t_y plane is physically impossible because it corresponds to contacts located at infinity relative to the origin.



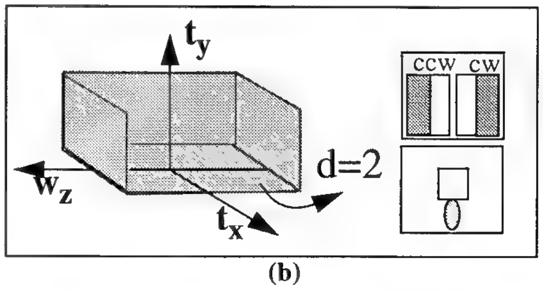


Figure 12 : r=1, d=2 Case. (a) A^* is just a plane. (b) A^* is a plane bounding a 3D space.

The *d-face* is a Line

When the rank of $\boldsymbol{A}$ is 2, the dimension of the *d-face* is $d=1$. The solution $\boldsymbol{A}^*$ according to the theorem can be either the linear subspace of dimension *1* or composed of faces with dimension *1* and *2*, or composed of faces of dimension *1*, *2*, and *3*.

Figure 13 shows the case when the solution $\boldsymbol{A}^*$ is a line. There are two cases corresponding to this $\boldsymbol{A}^*$: Figure 13(a) shows the case when the intersection of $\boldsymbol{A}^*$ with the t_x-t_y plane is a point. Figure 13(b) shows the case when the intersection is a line. The physical contact configurations corresponding to the two cases are kinematically different. One allows only rotation, while the other allows only translation.

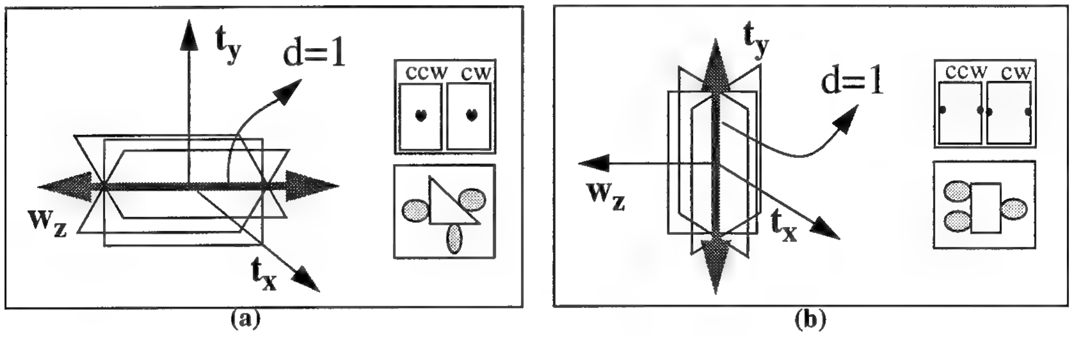


Figure 13 : r=2, d=1 Case. $\boldsymbol{A}^*$ is a line.

Figure 14 shows the case when the solution $\boldsymbol{A}^*$ is a PCC composed of a 2D half plane bounded by a line. Figure 14(a) shows the case when the intersection of $\boldsymbol{A}^*$ with the t_x-t_y plane is a half line. In this case, the object can only translate in one direction. Figure 14(b) shows the case when the intersection is a line. Here, the object can translate in the two directions of the line. The case when the half plane lies in the t_x-t_y plane is physically impossible because it corresponds to a contact at infinity.

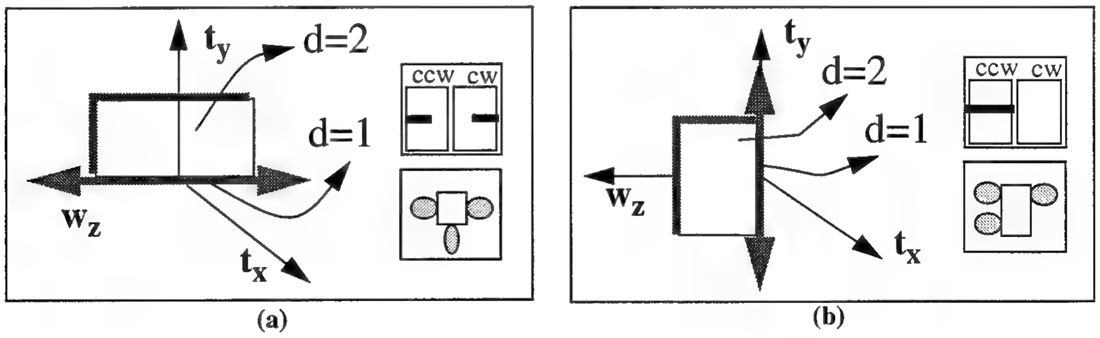


Figure 14 : r=2 and d=1 Case. $\boldsymbol{A}^*$ is a half plane bounded by a line.

When the solution $\boldsymbol{A}^*$ is a PCC composed of the 3D space bounded by two half planes which are bounded by a line, there can be three different ways of intersecting the t_x-t_y plane as shown in Figure 15. Figure 15(a) shows the case when $\boldsymbol{A}^*$ intersects the t_x-t_y plane in a line. The object in this case can translate in two directions. Figure 15(b) shows the case when the intersection is a half plane bounded by a line. Here, the object can translate in all directions on a semicircle. Figure 15(c) is the case when the intersection is a plane bounded by two half lines which are bounded by the origin. The object in this case can translate in all the directions bounded by the half lines.

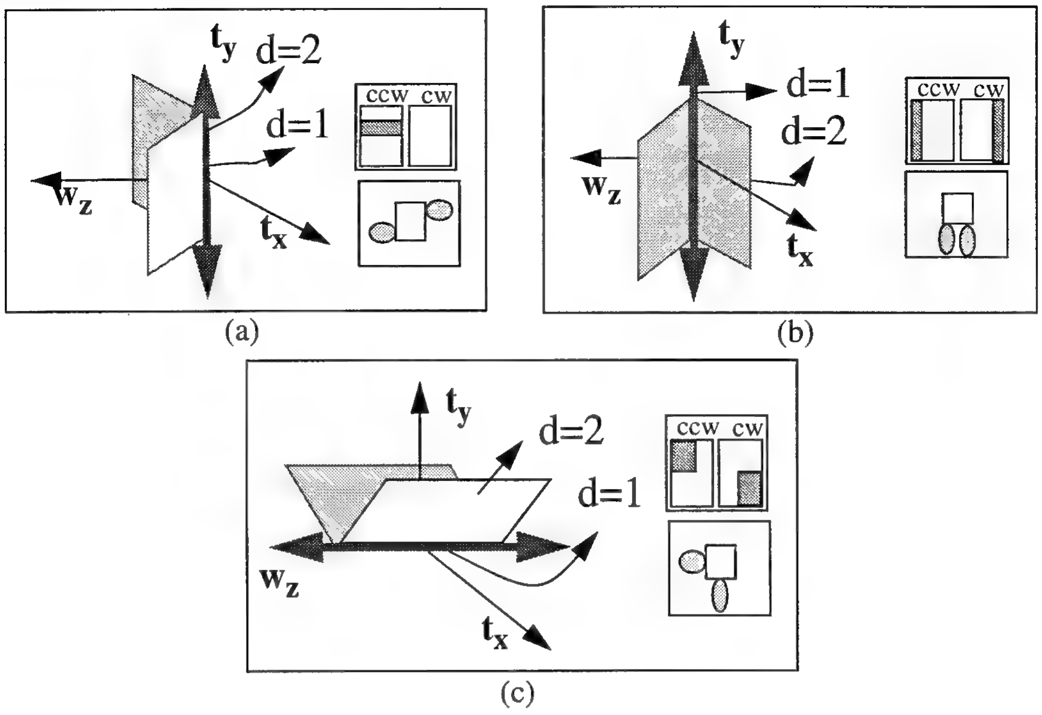


Figure 15 : r=2, d=1 Case: *A is the 3D space bounded by two half planes which are bounded by a line.**

The *d-face* is a Point

When the rank of *A* is *3*, the dimension of the *d-face d=0*. The solution *A** can be either the point (the origin) or a PCC composed of faces of dimension *0* and *1*, or a PCC composed of faces of dimension *0, 1,* and 2, or it could be a PCC composed of faces of dimension *0, 1,* 2 and *3.*

When *A** is just the linear subspace, the solution is just the origin (Figure 16). This is the case when there is no freedom of movement for the object in the plane.

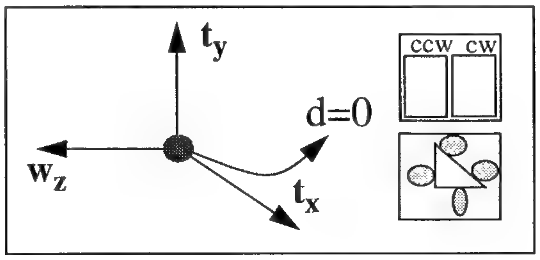


Figure 16 : r=3, d=0 Case. *A is a point.**

When *A** is composed of a half line bounded by the origin, two cases are possible depending on the intersection with the t_x-t_y plane. Figure 17(a) shows the case when the intersection of this *A** with the t_x-t_y plane is just a point. The object in this case cannot translate. Figure 17(b) shows the case when the intersection coincides with A^*, that is, a half line. Here the object can translate in only one direction.

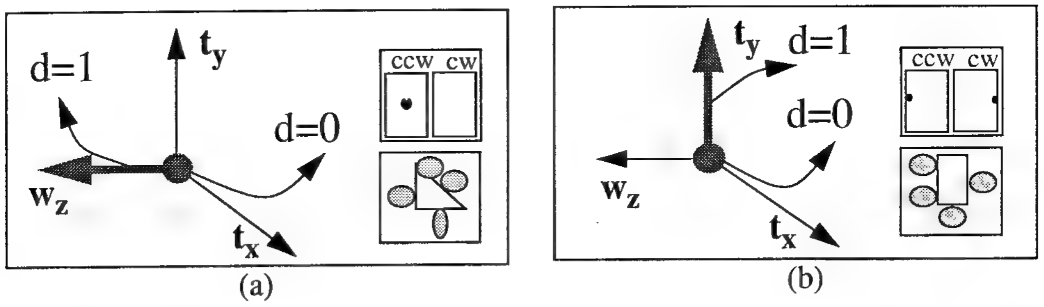


Figure 17 : r=3, d=0 Case. A^* a half line bounded by the origin.

Figure 18 shows the case when the solution A^* is a 2D plane bounded by two half lines which are bounded by the origin. Figure 18(a) shows the case when the intersection of A^* with the t_x-t_y plane is a point. The object in this case cannot translate. Figure 18(b) shows the case when the intersection is a half line, thus limiting the freedom of the object to translation in one direction. The case when the 2D face lies in the t_x-t_y plane is physically impossible because it corresponds to a contact at infinity.

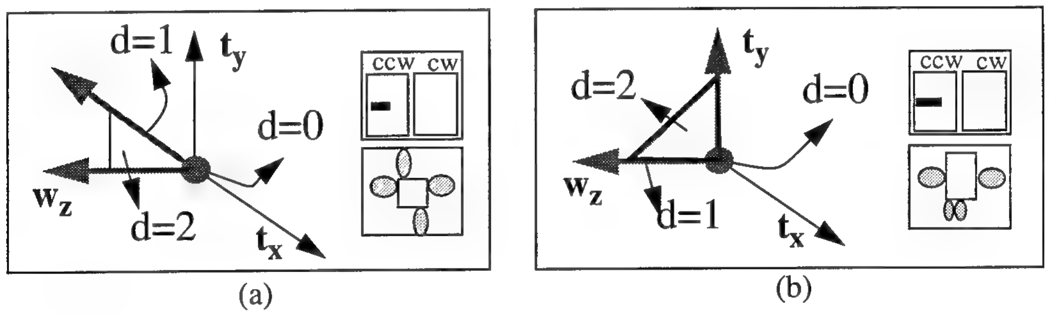


Figure 18 : r=3, d=0 Case. A^* is a 2D plane bounded by half lines which are bounded by the origin.

Another combination of faces making up the solution A^* is the 3D space bounded by planes which are bounded by half lines which are in turn bounded by the origin (Figure 19). Figure 19(a) shows the case when A^* intersects the t_x-t_y plane in a point. The object cannot translate in this case. Figure 19(b) shows the case when the intersection is a half line bounded by the origin. The object in this contact configuration can only translate in one direction. Figure 19(c) shows the case when the intersection is the 2D t_x-t_y plane bounded by two half lines which are bounded by the origin. The object in this case can translate along all directions bounded by these half lines.

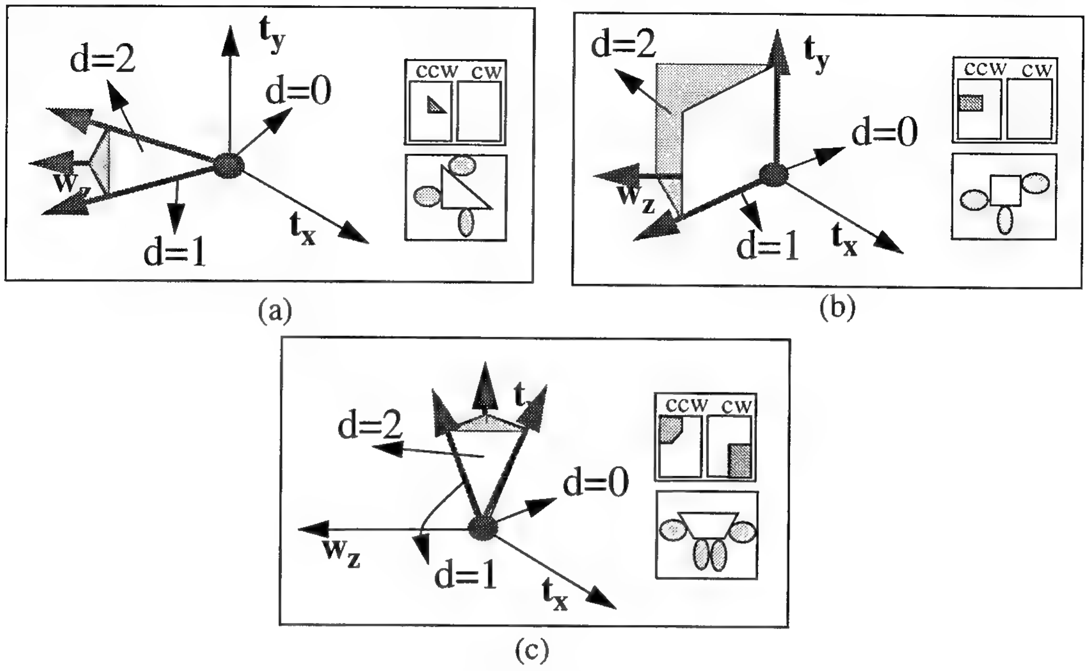


Figure 19 : r=3, d=0 Case. A^* is a 3Dspace bounded by planes which are bounded by half lines which are bounded by the origin.

The summary of the analysis is shown in Figure 20. There are 18 contact states. Any contact configuration can be classified into one of these eighteen states. The PCC theory guarantees that these constitute the complete set of contact states.

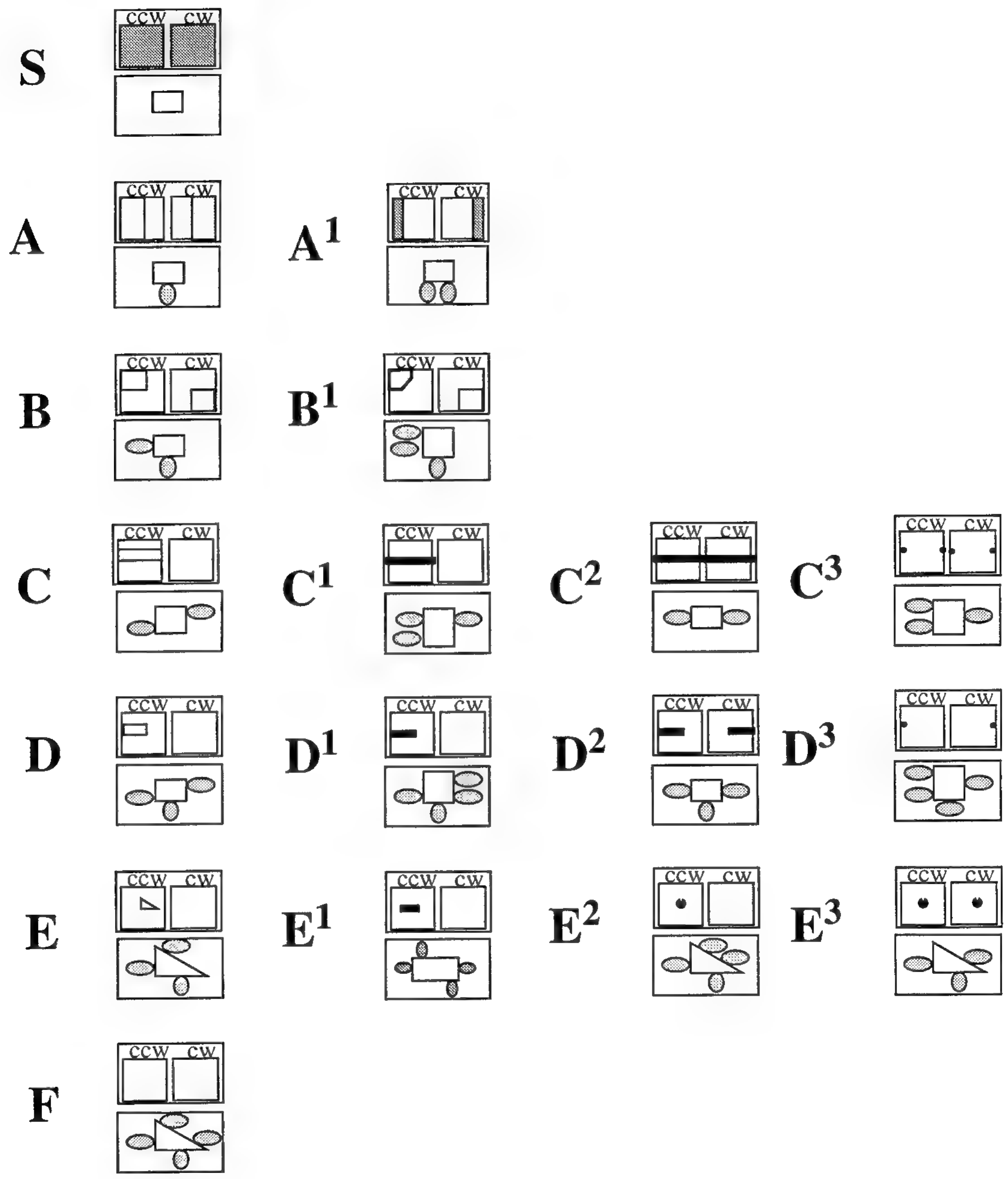


Figure 20 : Assembly State Classification for 2D Translation and Rotation.

The topology of the solution A* and its intersection with the t_x-t_y plane can all be concisely represented as a table as shown in TABLE 1. The six states for the 2D translation case and the 10 states for the 3D translation case are simply special cases.

Rank r	**d = 3-r**	**Dimension of faces of the PCC *A****	**Dimension of faces in the t_x-t_y plane**
0	3	3	2
1	2	2	1
1	2	2,3	1,2
2	1	1	1
2	1	1	0
2	1	1,2	1
2	1	1,2	0,1
2	1	1,2,3	1
2	1	1,2,3	1,2
2	1	1,2,3	0,1,2
3	0	0	0
3	0	0,1	0
3	0	0,1	0,1
3	0	0,1,2	0
3	0	0,1,2	0,1
3	0	0,1,2,3	0
3	0	0,1,2,3	0,1
3	0	0,1,2,3	0,1,2

TABLE 1. Combinatorial Logic of Classification.

Each contact state can be represented by a minimum number of contacts. The eighteen states can thus be sorted by the minimum number of contacts as shown in Figure 21. This is useful for building the transition graph which will be explained in the following section.

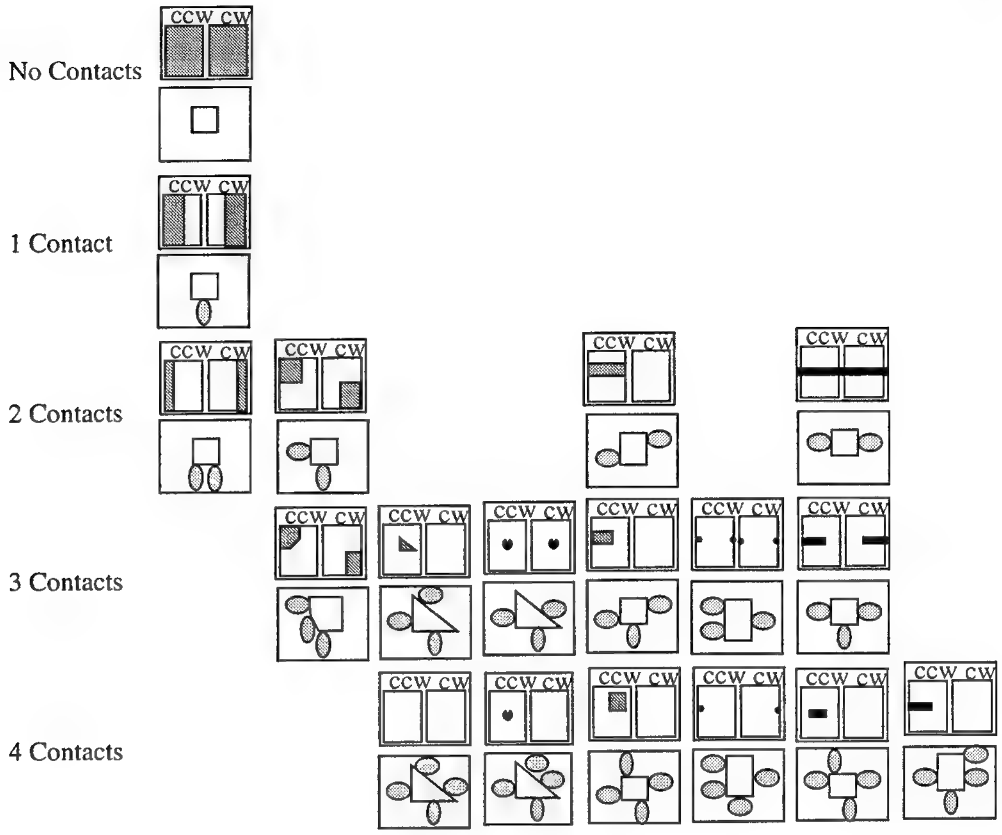


Figure 21 : Contact States sorted by the number of contacts

3.3 Transition Graph

The freedom derived from the lines of contact represents instantaneous freedom. Some of the legal motions will change the contact state. The transition graph is a representation of all the possible changes from one contact state to another, whether due to translation or to rotation.

We assume that an assembly motion is a finite sequence of simple motions (pure translations or pure rotations) between contact states. Thus the assembly actions can be represented as a path on the transition graph. Each transition achieves a combination of the following:

•Establishing a new contact.

•Maintaining a previous contact.

•Detaching a previous contact.

Using the contact states and the transition motions we can model what the operator achieved, during an assembly.

3.3.1 Transition Graph for Polygonal Objects

The transition graph for polyhedral objects is shown in Figure 22. Graph edges marked T indicate that translation can cause the transition and R indicates that rotation can cause the transition. There are 28 transitions possible between the 17 states. (The F state does not allow any freedom, so is not included in the transition graph.) The transition motions are chosen to be pure translations or pure rotations that decrease the number of contacts (according to the sorted states) gradually if possible.

When we are dealing with polygonal objects and if rotation is involved, only the contacts with normals going through the rotation center will be maintained. This implies that more than one vertex on line contact cannot be maintained in a rotation. Thus, rotations of polygonal objects can only cause transitions from that state to the A or S states.

When a sequence of observed states is available, the path from each state to the next can be traced through the transition graph. When both translation and rotation can cause a transition, then the observation is used to choose of the type of motion used by the operator.

It must be noted that in contrast to conventional task planning approaches like C-space planning, the aim of the transition graph is to provide a framework for efficiently acquiring the necessary information from the observations.

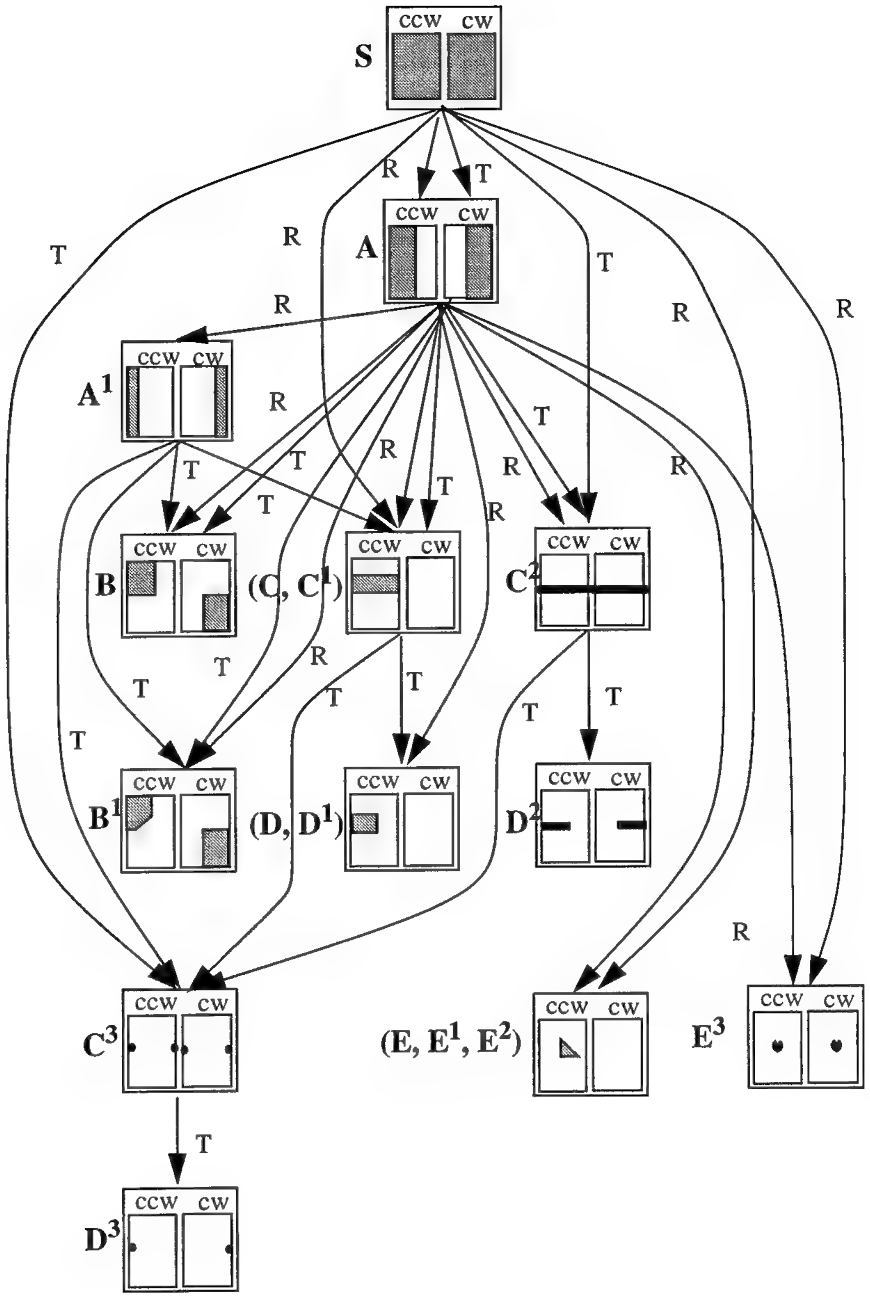


Figure 22 : Transition Graph for Polygonal Objects

3.4 Motion Macros

A motion macro is the data structure holding all the information about the action needed to change from the previous assembly state to the next assembly state. This information can be directly converted into manipulator actions. We assume actions are simple motions, like pure translation or pure rotation of finite magnitude. To completely define a motion macro, we need the parameters that are required for executing the motion.

All translation motion macros contain the following parameters:

•Direction of translation

•Magnitude of translation

•Final position (or initial position)

All rotation motion macros contain the following parameters:

•Direction of rotation (clockwise or counterclockwise)

•Center of rotation

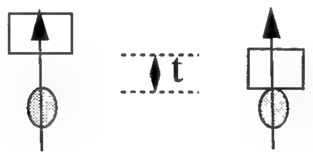
•Magnitude of rotation

•Final position (or initial position)

The motion macros indicate the manipulator motions needed to achieve an assembly state transition. In cases where the robot manipulator actions are extremely accurate, pure position control may be able to achieve the task. In cases where there is uncertainty, the manipulator motion will have to be controlled using both position and force (hybrid) control[15]. In these cases, the motion macros must indicate the coordinates along which force control is needed and those along which position control is appropriate. This kind of skill-based manipulation is implemented in [22].

There are 28 transitions between the 17 assembly states. The motion macros for each of them is described in Appendix A. Two representative motion macros are given below.

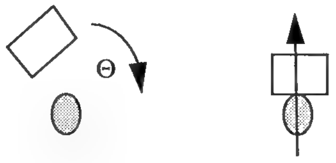
S-to-A (Translate to touch 1) T1



Translate to touch 1: S-A

The motion is Translation. The direction is opposite to the normal of contact. The magnitude (t) of translation is obtained from observation. If force control is available during execution, the contact can be sensed as a force along the contact normal.

S-to-A (Rotate to touch 1) R1



Rotate to touch 1: S-A

The motion is rotation. The direction of rotation, center of rotation, and magnitude of rotation are obtained from observation. If force control is available during execution, then the torque about the rotation center can be used to sense the contact.

3.5 Procedure Graph

The procedure graph is the transition graph with the motion macros attached to the corresponding edges of the graph (see Figure 23). The procedure graph is used by the task instantiation module to convert the abstract task model into manipulator commands.

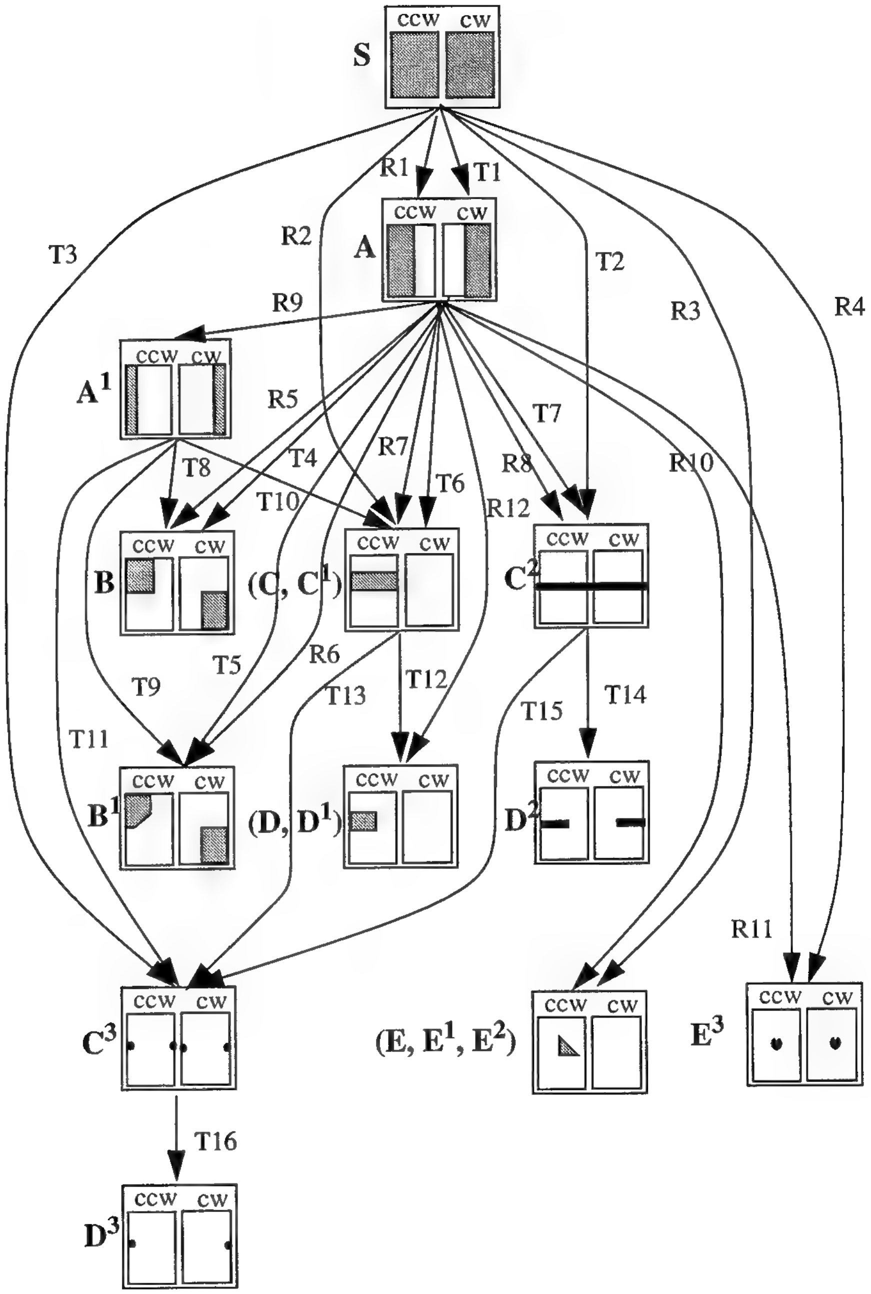


Figure 23 : Procedure graph for 2D polygons with translation and rotation.

3.6 Task Instantiation

Once the assembly states of the manipulated object are determined, and the necessary assembly actions are understood, the task instantiation module generates the necessary robot actions to repeat the assembly task in the robot workspace. The pre-assembly state and post-assembly state for the manipulated object together with the procedure graph provide the motion macros for accomplishing the task. The instantiated task will consist of a series of manipulator motions which will accomplish the task in the robot's workspace.

All the previous theory deals with only one subtask of an observed assembly task. This is the final assembly of the manipulated object. To execute an assembly task involving a newly manipulated object, we need the instantiation of the following subtasks in this order:

1. The grasp strategy for gripping the manipulated object. This can be obtained from observation [8]. Our implementation uses predefined gripping points for each object.

2. The disassembly of the manipulated object from the robot's parts-table. This task can also be observed if we can observe the operator disassembling the object from a parts-table. But for practical reasons, the objects are stored in known configurations on the parts-table. The disassembly path for removing them from the parts-table is obtained from the procedure tree by reversing the paths of the transitions.

3. The global path that is needed for the transition from the S state after disassembly from the parts-table to the S state at the work-table before the assembly. This subtask can also be obtained from observation. But for practical reasons a simple collision free path planning approach is used for implementation.

4. The assembly of the manipulated object from the S state to the final state in the assembly obtained from the observation as explained in the previous sections.

Once all these subtasks are instantiated, the robot commands are sent to the robot, which then executes the subtasks in the same order.

4 Implementation

We have implemented the APO system for the case of polyhedral objects assembled using both rotation and translation. The structure is similar to that of the system which deals with polyhedral objects assembled with translation only[23].

4.1 Observation of the assembly

The polygonal parts used for observation are lightly colored objects. They are assembled in front of a monochrome camera in a black background. The stages of assembly corresponding to each of the subtasks are then recorded as shown in Figure 24. (This temporal segmentation can be done automatically [9], but is done manually here)

Each image is subtracted from its predecessor, and thresholded. The resulting regions are fitted with polygons, which is then used for recognizing the manipulated object. The result is the model and configuration of the manipulated object used in the subtask.

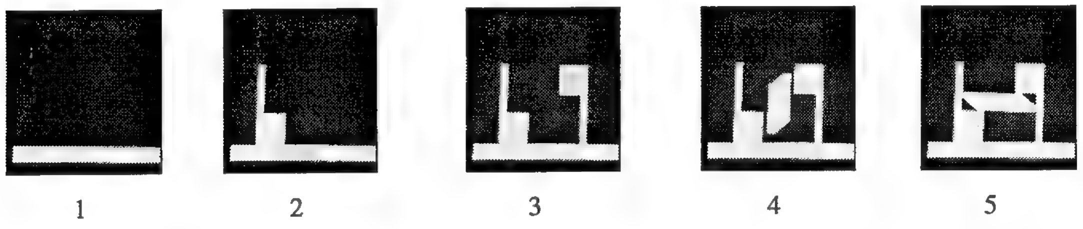


Figure 24 The Observed Assembly Sequence.

The parts used in the assembly are modelled using the geometric modelling system VANTAGE [2]. The polygonal objects are modelled as flat 3D objects in VANTAGE. A graphical representation of the final observation shown in Figure 24(5) is shown in Figure 25.

The observation can only provide a rough configuration of the object. To correct any error in the observed configurations, the APO system uses fine localization based on face contact constraints [24]. Fine localization is important in finding the true contacts and computing the assembly state.

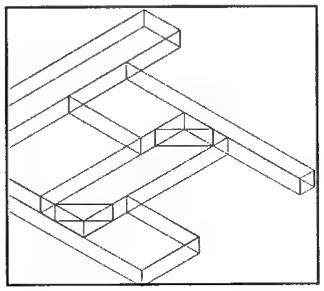


Figure 25 Models of the final assembly in VANTAGE

4.2 Task Recognition

The task recognition module is built using the theory explained in Section 3 The process involves building the following: The transition graph, the motion macros, the procedure graph, the functions which compute the contact state and the functions for finding the path in the transition graph, the functions to find the gripping point and the functions to find a collision free path between two points.

In order to simplify the implementation we assume that the operator picks up the manipulated object from a known location and then transfers it to the worktable and assembles it into the environmental objects there.

When the APO system is in operation, the observation module provides the identity and configuration of manipulated object. Using the VANTAGE models, pairs of the object features and the environment features are checked for contact. The geometry of all the contacts are used to compute the contact state of the manipulated object.

Once the state is computed, the transitions necessary to reach the S state or the previous state is found by searching the transition graph. The path is stored as the sequence of transitions needed to get to the S state. For example, in the case of the assembly in Figure 24, the L-shaped object is in state A^1. The transition path is obtained as a S-to-A^1 translation for the assembly of the L-shaped part at the work-table.

There are three transition paths involved in a stage of the assembly. For example, for the case of the assembly of the L-shaped part in Figure 24(2), it involves the following: The transition from A^1-to-S at the parts-table, the S-to-S transition during the motion from the parts-table to the work-table, and the transition from S-to-A^1 at the work-table. Once all three transitions are obtained, they are concatenated to form a basis for the assembly plan for accomplishing the subtask.

Since the emphasis of the work is not on grasp recognition, the grasping of the manipulated object is simplified by using predetermined gripping positions for each of the objects used in the assembly. In cases where there are multiple gripping points, a gripping point that does not interfere with other objects during the assembly is used when picking up the manipulated object from the parts table.

A collision free path is needed to get the manipulated object from the parts-table to the vicinity of the work-table. A simple path planning function is used to avoid obstacles.

4.3 Task Instantiation

The task is instantiated by obtaining the motion macros corresponding to the transitions from the procedure graph. For the example of the L-shaped object, the motion macros for the three transitions A^1-to-S, S-to-S, and S-to-A^1 are obtained. The corresponding motion macros are translate-to-detach, move, and translate-to-touch respectively.

The values of the parameters needed for each of these motion macros are then obtained by computation or from intermediate observations. The need for intermediate observations is particularly useful in the case of the rotation motions involving polygonal objects

Intermediate Observations

The present APO system relies totally on one final observation in each subtask of the assembly process. This is not sufficient in cases where the parameters of the motion such as the angle of rotation and the distance of translation are important. In these cases the intermediate observations can be used to compute the values of the necessary parameters. For example, in the assembly shown in Figure 24, the rotation angle was computed using the final two observations (4 and 5).

If the transition motion is rotation, the feasible rotation centers can be obtained from the solution of the screw inequalities. The rotation center could lie in an area on the 2D plane but the APO system uses the one employed by the human operator. This can be obtained from intermediate observations. For the ease of implementation, we choose the rotation center as the center of the feasible rotation area.

When both rotation and translation are possible from a state, then we need to recognize which is being used by the operator. From consecutive observations, it is possible to decide if the motion was a rotation or a translation. In the present implementation, we assume that, when both rotation and transition are possible for a transition, then translation is chosen.

4.4 Task Execution

The instantiated task is used to program the Manipulator-1 inside RobotWorld[3]. The manipulator commands are generated from each motion macro. The generated program for the assembly of a part involves commands to pick it up at the parts-table, move it over to the work-table, and assemble it at the worktable. An example is shown in Figure 26

```
OPEN(1)
MOVES(1,[2.850, 10.965])
ROTATES(1,[2.850, 10.965, 2.638, 0.000])
APPROACHS(1,[2.850, 10.965, 2.638, 0.000])
MOVES(1,[2.850, 10.965])
ROTATES(1,[2.850, 10.965, 1.080, 0.000])
APPROACHS(1,[2.850, 10.965, 1.080, 0.000])
CLOSE(1)
ROTATES(1,[2.850, 11.358, 1.080, 0.000])
MOVES(1,[2.850, 11.358])
APPROACHS(1,[2.850, 11.358, 1.080, 0.000])
```

Disassembly at the parts-table.

```
MOVES(1,[2.850, 11.358])
MOVES(1,[2.850, 15.295])
```

Global Motion to the work-table.

```
MOVES(1,[24.307, 11.516])
MOVES(1,[24.307, 7.579])
ROTATES(1,[24.307, 7.579, 1.080, 0.000])
APPROACHS(1,[24.307, 7.579, 1.080, 0.000])
MOVES(1,[24.307, 7.185])
ROTATES(1,[24.307, 7.185, 1.080, 0.000])
APPROACHS(1,[24.307, 7.185, 1.080, 0.000])
OPEN(1)
```

Assembly at the work-table.

Figure 26 Generated RobotWorld commands for an instantiated assembly task

The assembly task observed in Figure 24, is executed in three stages inside RobotWorld workspace as explained below.

The snapshots during the assembly of part-1 is shown in Figure 27. Figure 27(a) shows the disassembly of part-1 from the parts-table. The transition is from the state A^1 to state S. The manipulator then moves part-1 from the state S at the parts-table to the state S at the work-table. Figure 27(b) shows the assembly of part-1 at the work-table. The transition is from state S to state A^1.

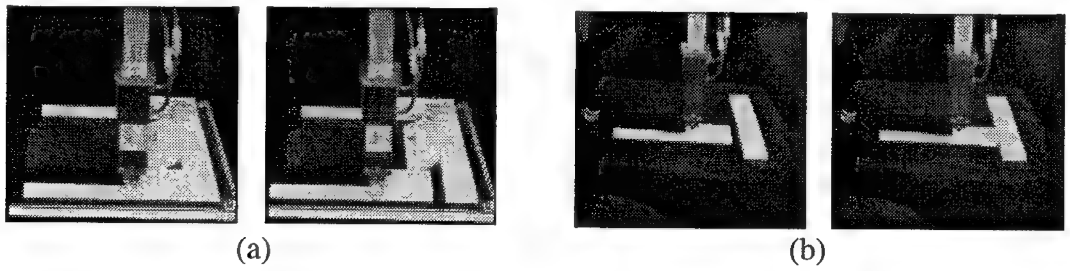


Figure 27 : (a) A^1-to-S transition for part1 (b) S-to-A^1 transition for part1

3. RobotWorld is a multi manipulator robot system. The manipulators have four degrees of freedom. The three degrees of freedom are cartesian along the XYZ axes. The end-effector has a rotational degree of freedom about the Z axis. The actuators are stepper motors with an accuracy of 0.01 inch.

The stages during the assembly of part-2 is shown in Figure 27. Figure 27(a) shows the disassembly of part-2 from the parts-table. The transition is from the state A^1 to state S. The manipulator moves part-2 from the parts-table to the work-table. Then, part-2 is assembled at the work-table as shown in Figure 27(b). The transition here is from state S to state A^1.

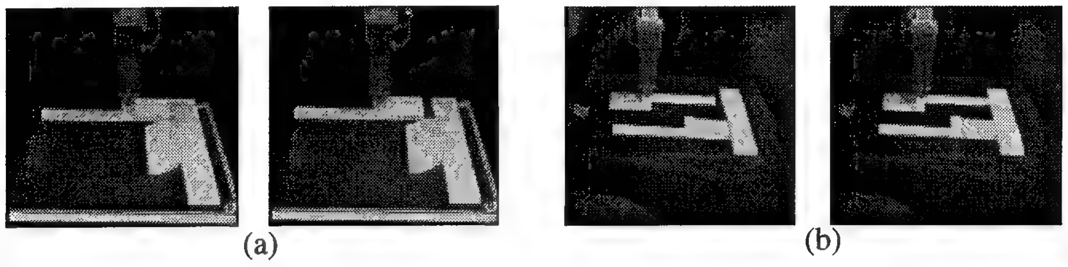


Figure 28 : (a) A^1-to-S transition for part2 (b) S-to-A^1 transition for part2

The case for part-3 is shown in Figure 29. Figure 29(a) shows the disassembly of part-3 from the parts-table. The transition is from the state A^1 to state S. Figure 29(b) shows the assembly of part-3 at the work-table. The transition is from state S to state E. This is the stage in the assembly task (Figure 24 (4)) where the intermediate observation is used to obtain the angle of rotation.

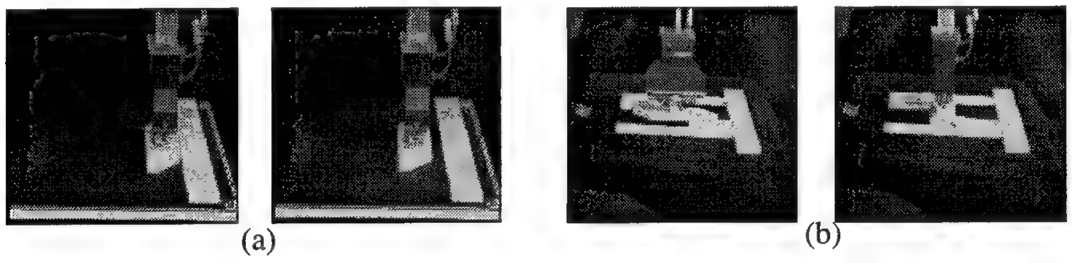


Figure 29 : (a) A^1-to-S transition for part3 (b) S-to-E transition for part3

The current task instantiation module has motion macros which assume no uncertainty in the execution module. Thus the motion macros hold only position information. The task execution is also done in the open loop mode. This is sufficient for the example shown, because of the high accuracy of the RobotWorld manipulators. This may not be suited for more precise assemblies. In these cases, we plan to introduce feedback to reliably execute the assembly task.

Conclusion

We have successfully used polyhedral convex cone theory to explain the basis for partitioning contact space into a finite number of contact states. The theory now guarantees completeness.

We have built and implemented the APO system for observing planar assemblies of polygonal objects, where the assembly motions include rotation, in addition to translation. This involved classifying the assemblies into states using screw theory, finding the transition graph, the procedure graph and the implementation of the system using a vision system and the Robotworld.

The implementation brought into focus the need of intermediate observations and its use for computing the parameters for the motion macros. A high speed perception system might be needed for implementation.

The screw theory that forms the basis for the representation of freedom can represent motion in 3D space. The polyhedral convex cone theory can be the mathematical basis for the classification of the contact states in 3D. With these tools we hope to extend the system for the case of 3D objects with rotation and translation freedom in 3D space. We also plan to explore sensor-based manipulation during task execution to improve reliability.

Acknowledgments

We wish to thank Mark Wheeler, Sing Bing Kang, Jutta Denton, Fred Solomon, and Martial Hebert for their help in preparing this report.

References

[1] Ball, T.S. A Treatise on the Theory of Screws. Cambridge University Press (1900).

[2] Balakumar, J.C. Robert, R.H. Ikeuchi, K. and Kanade, T. Vantage: A Frame-based Geometric/Sensor Modelling System - Programmer/User's Manual v1.0. Technical report, Carnegie Mellon University, Robotics Institute, 1989.

[3] Finkel, R. Taylor, R. Bolles, R.P. and Feldman, J. AL, a Programming System for Automation. Technical Report AIM-177, Stanford University, Artificial Intelligence Laboratory, Stanford, CA, 1974.

[4] Hirai, S. and Sato, T. Motion Understanding for World Model Management of Telerobot. In *IEEE Inter. Conf. on Intelligent Robots and Systems,* pages 124-131, 1989.

[5] Hirai, S, Asada, H, and Tokumaru, H. Kinematic Analysis of contact state transitions in Assembly operations and automatic generation of transition networks. *Journal of the Robotics Society of Japan*, 1987, vol 24, no 4. pages 84-91 (in japanese).

[6] Hirai, S, and Iwata, K. Recognition of Contact State Based on Geometric Model. *IEEE Int Conf on R & A*, 1992, vol 2, pages 1507-1512.

[7] Hunt, K.H. Kinematic Geometry of Mechanisms, Clarendon, Oxford (1978).

[8] Kang S.B. and Ikeuchi, K. Towards Automatic Robot Instruction from Perception: Recognizing a Grasp from Observation. IEEE Trans on R&A, 1993, vol 9, no 4, pages 432-443.

[9] Kang S.B. and Ikeuchi, Determination of Motion Breakpoints in a Task Sequence from Human Hand Motion. Prc IEEE Int Conf on R&A, 1994, pages 551-556.

[10] Kuhn, H.W. and Tucker, A.W. editors, Linear Inequalities and Related Systems, Annals of Math Studies, Vol 39, Princeton, 1956.

[11] Lammineur, P. and Cornillie, O. Industrial Robots, Pergammon press, pages 43-54, 1984.

[12] Laugier, C. Planning fine motion strategies by reasoning in the contact space. *IEEE Int Conf on R & A* 1989, pages 653-659.

[13] Lieberman, L.L. and Wesley, M.A. Autopass: an Automatic Programming System for Computer Controlled Mechanical Assembly. *IBM Journal of Res. and Develop.* 21(4):321-333, 1977.

[14] Lozano-Perez, T. Automatic Planning of Manipulator Transfer Movements. *IEEE Trans. System Man and Cybernetics,* SMC-11(10):681-689, 1981.

[15] Lozano-Perez, T. Mason, M.T. and Taylor, R.H. Automatic Synthesis of Fine-motion Strategies for Robots. In M. Brady and R. Paul, editors, Robotics Research 1, pages 65-96. MIT Press, Cambridge, MA 1984.

[16] Lozano-Perez, T. Automatic Planning of Manipulator Transfer Movements. *IEEE Trans System Man and Cybernetics,* SMC-11(10):681-689, 1981.

[17] Lozano-Perez, T. and Winston, P.H. Lama: a Language for Automatic Mechanical Assembly. In *Proc. of 5th Intern. Joint Conf. on Artificial Intelligence*, pages 710-716, 1977.

[18] Mattikalli, R S, and Khosla, P K. Motion Constraints from Contact Geometry: Representation and Analysis. *IEEE Int Conf on R & A*, 1992, vol 3, pages 2178-2185.

[19] Ohwovoriole, M.S. and Roth, B. An Extension of Screw Theory. *Journal of Mechanical Design,* Oct 1981

[20] Popplestone, R.J. Ambler, A.R and Bellos, I. An Interpreter for a Language for Describing Assemblies. *Artificial Intelligence,* 14(1): 79-107, 1980.

[21] Silwa, N.O. and Will, R.W. A Flexible Telerobotic System for Space Operations. In *Proc. Space Telerobotics Workshop,* pages 285-292, Pasadena, 1987.

[22] Suehiro, T. and Taksase, R. Skill Based Manipulation System. *Journal of the Robotics Society of Japan*, 8(5):47-58, October 1990. (in Japanese).

[23] Ikeuchi, K. and Suehiro, T. Towards an Assembly Plan from Observation, part i: Assembly Task Recognition using Face Contact Relations (Polyhedral Objects). In *Proc of IEEE Intern Conf on Robotics and Automation*, Nice France, May 1992.

[24] Suehiro, T. and Ikeuchi, K. Towards an Assembly Plan from Observation: Fine Localization Based on Face Contact Constraints. Technical Report, CMU-CS-91-168. August 1991.

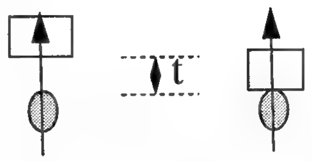
[25] Wilson, R.H. and Matsui, T. Partitioning an Assembly for Infinitesimal Motions in Translation and Rotation. Proc. of the 1992 *IEEE Int. Conf. on Intelligent Robots and Systems.* Raleigh, NC, July 1992.

[26] Whitney, D.E. State Space Models of Remote Manipulation Tasks. In *Proc. of 1st Intern, Conf. Artificial Intelligence*, pages 495-507, 1969.

Appendix A

A motion macro is a template holding the information needed for a manipulator to perform an assembly task. The information in a template will depend on the type of action performed and on the manipulator. There are 24 transitions between the 12 assembly states as shown in the procedure graph (Figure 23). The motion macros for each of them are described as follows.

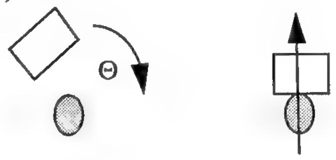
A.1 S-to-A (Translate to touch 1) T1



Translate to touch 1: S-A

The motion is translation. The direction is opposite to the normal of contact. The magnitude (t) of translation is obtained from observation.

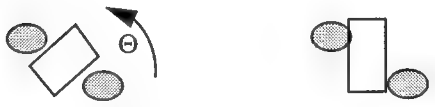
A.2 S-to-A (Rotate to touch 1) R1



Rotate to touch 1: S-A

The motion is rotation. The direction of rotation, center of rotation, and magnitude of rotation are obtained from observation.

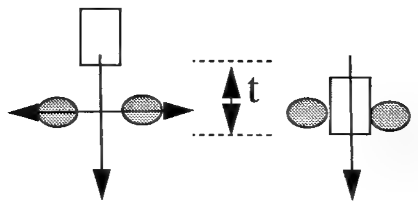
A.3 S-to-C (Rotate to touch 2) R2



Rotate to touch 2: S-C

The motion is rotation. The direction of rotation, center of rotation, and magnitude of rotation are obtained from observation.

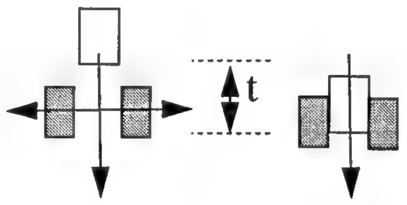
A.4 S-to-C' (Translate to insert 1) T2



Translate to insert 1: S-C'

The motion is translation.The direction is perpendicular to the normals of contact.The magnitude (t) of motion are obtained from observation.

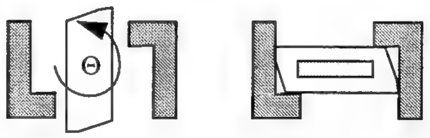
A.5 S-to-C" (Translate to insert 2) T3



Translate to insert 2: S-C"

The motion is translation.The direction is perpendicular to the normals of contact.The magnitude (t) of motion are obtained from observation.

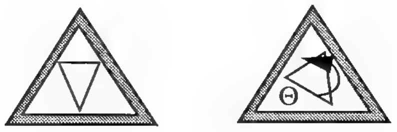
A.6 S-to-E (Rotate to touch 3) R3



Rotate to touch 3: S-E

The motion is rotation. The direction of rotation, center of rotation, and magnitude of rotation is obtained from observation.

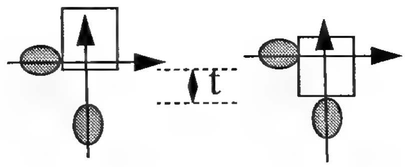
A.7 S-to-E' (Rotate to touch 4) R4



Rotate to touch 5: S-E'

The motion is rotation. The direction of rotation, center of rotation, and magnitude of rotation is obtained from observation.

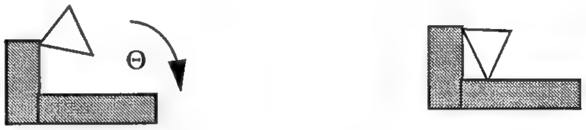
A.8 A-to-B (Translate to touch 2) T4



Translate to touch 2: A-B

The motion is translation. The direction is opposite to the normal of the new contact. The magnitude (t) of translation is obtained from observation.

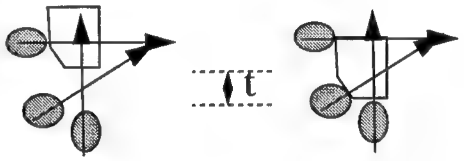
A.9 A-to-B (Rotate to touch 5) R5



Rotate to touch 5: A-B

The motion is rotation. The direction of rotation, center of rotation, and magnitude of rotation is obtained from observation.

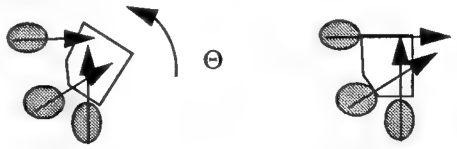
A.10 A-to-B' (Translate to touch 3) T5



Translate to touch 3: A-B'

The motion is translation. The direction is perpendicular to the normal of the previous contact. The magnitude (t) of translation is obtained from observation.

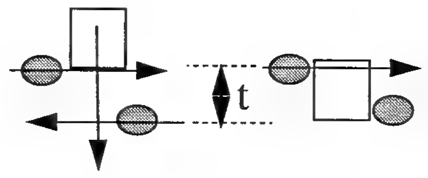
A.11 A-to-B' (Rotate to touch 6) R6



Rotate to touch 6: A-B'

The motion is rotation. The direction of rotation, center of rotation, and magnitude of rotation is obtained from observation.

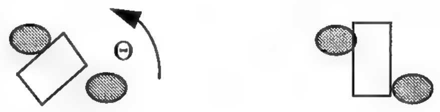
A.12 A-to-C (Translate to insert 3) T6



Translate to insert 3: A - C

The motion is translation.The direction is perpendicular to the normal of the new contact.The magnitude of (t) translation is obtained from observation.

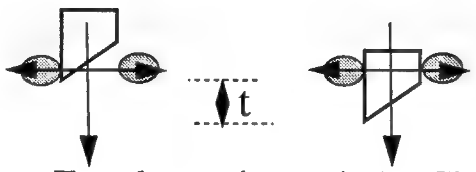
A.13 A-to-C (Rotate to touch 7) R7



Rotate to touch 7: A-C

The motion is rotation. The direction of rotation, center of rotation, and magnitude of rotation is obtained from observation.

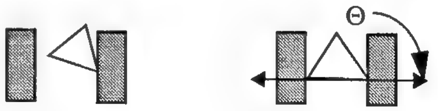
A.14 A-to-C' (Translate to insert 4) T7



Translate to insert 4: A - C'

The motion is translation.The direction is perpendicular to the normal of the new contact.The magnitude of (t) translation is obtained from observation.

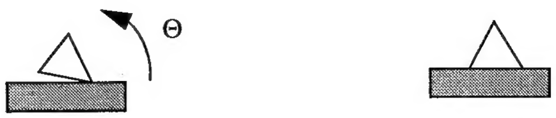
A.15 A-to-C' (Rotate to touch 8) R8



Rotate to touch 8: A-C'

The motion is rotation. The direction of rotation, center of rotation, and magnitude of rotation are obtained from observation.

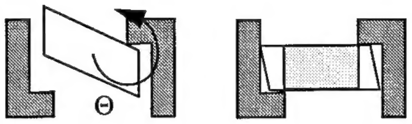
A.16 A-to-A^1 (Rotate to touch 9) R9



Rotate to touch 9: A-A^1

The motion is rotation. The direction of rotation, center of rotation, and magnitude of rotation are obtained from observation.

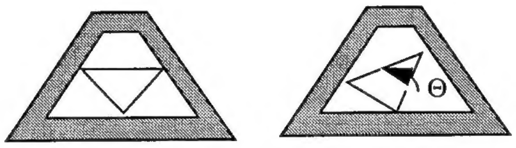
A.17 A-to-E (Rotate to touch 10) R10



Rotate to touch 10:A-E

The motion is rotation. The direction of rotation, center of rotation, and magnitude of rotation are obtained from observation.

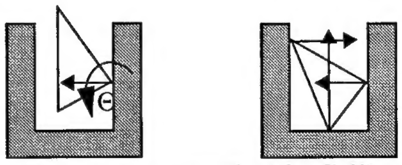
A.18 A-to-E' (Rotate to touch 11) R11



Rotate to touch 11: A-E'

The motion is rotation. The direction of rotation, center of rotation, and magnitude of rotation are obtained from observation.

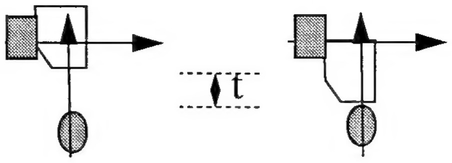
A.19 A-to-D (Rotate to touch 12) R12



Rotate to touch 12: S-D

The motion is rotation. The direction of rotation, center of rotation, and magnitude of rotation are obtained from observation.

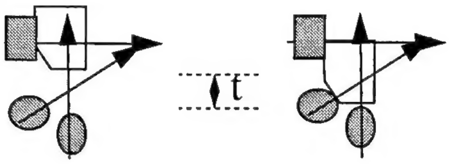
A.20 $A^1$to-B (Translate to touch 4) T8



Translate to touch 4: A^1-B

The motion is translation. The direction is perpendicular to the normal of the previous contact. The magnitude (t) of translation is obtained from observation.

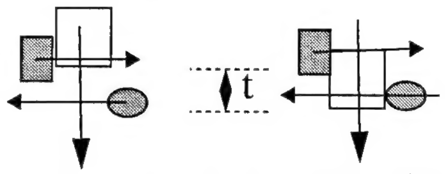
A.21 $A^1$to-B' (Translate to touch 5) T9



Translate to touch 5: A^1-B'

The motion is translation. The direction is perpendicular to the normal of the previous contact. The magnitude (t) of translation is obtained from observation.

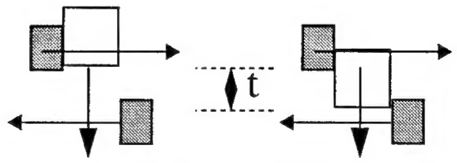
A.22 A^1-to-C (Translate to insert 5) T10



Translate to insert 5: A^1-C

The motion is translation. The direction is perpendicular to the normal of the new contact. The magnitude (t) of translation is obtained from observation

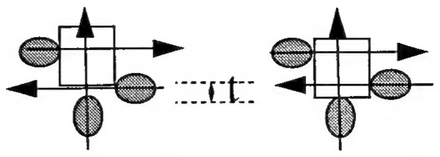
A.23 A^1-to-C" (Translate to insert 6) T11



Translate to insert 6: A^1-C"

The motion is translation. The direction is perpendicular to the normal of the new contact. The magnitude (t) of translation is obtained from observation

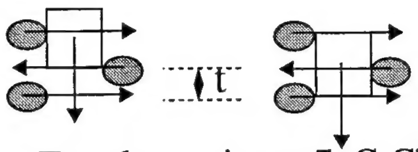
A.24 C-to-D (Translate to touch 6) T12



Translate to touch 6: C-D

The motion is translation. The direction is opposite to the normal of the new contact. The magnitude (t) of translation is obtained from observation.

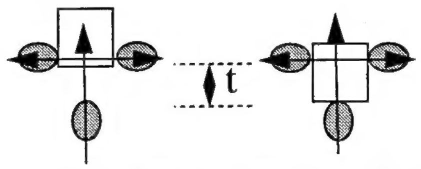
A.25 C-to-C" (Translate to insert 7) T13



Translate to insert 7: C-C"

The motion is Translation. The direction is perpendicular to the normal of the new contact. The magnitude (t) of translation is obtained from observation

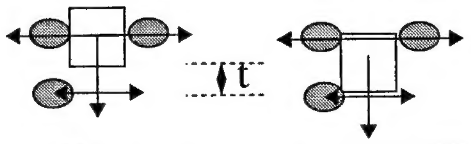
A.26 C'-to-D' (Translate to touch 7) T14



Translate to touch 7: C'-D'

The motion is translation. The direction is opposite to the normal of the new contact. The magnitude (t) of translation is obtained from observation.

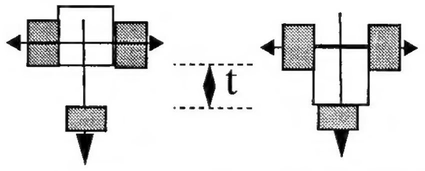
A.27 C'-to-C" (Translate to insert 8) T15



Translate to insert 8: C'-C"

The motion is Translation. The direction is perpendicular to the normal of the new contact. The magnitude (t) of translation is obtained from observation

A.28 C”-to-D” (Translate to touch 8) T16



Translate to touch 8: C”-D”

The motion is translation. The direction is opposite to the normal of the new contact. The magnitude (t) of translation is obtained from observation.